A New Trapdoor in Modular Knapsack Public-Key Cryptosystem

Takeshi Nasako *　　Yasuyuki Murakami †

Abstract. Merkle and Hellman proposed a first knapsack cryptosystem. However, it was broken because the density is not sufficiently high. In this paper, we propose a new trapdoor in modular knapsack PKC. This trapdoor uses some products of bases. We also propose a new modular knapsack PKC based on proposed trapdoor. Moreover, we discuss security against the exhaustive search and the low-density attack.

1 Introduction

The realization of the quantum computer will enable to break public-key cryptosystems based on factoring problem and discrete logarithm problem [1)]. Under this future threat, it is important to search for secure PKCs based on the other problem. The knapsack problem is one of the remarkable problems to apply to design PKC.

The knapsack problem is to find the solution $(x_1, x_2, \ldots, x_n) \in \{0,1\}^n$ such that

$$C = a_1x_1 + a_2x_2 + \cdots + a_nx_n$$

for given positive integers $a_1, a_2, \ldots, a_n$ and some of these sum C. This problem is known to be *NP*-hard. This fact has motivated us to invent a public-key cryptosystem based on the knapsack problem.

Merkle and Hellman proposed the first knapsack public-key cryptosystem(MH PKC) [2)]. MH PKC has a remarkable feature that the encryption and the decryption can be performed very fast. However, it is known that the basic MH PKC can be broken by Shamir's attack [3)] and Adleman's attack [4)] because it uses a superincreasing sequence in the trapdoor. It is also broken with the low-destiny attack [5, 6)] because the density is low. These attacks have given the impression that knapsack PKCs are insecure. It is, however, difficult to condemn that all knapsack PKCs can not be secure.

The density, an important parameter in knapsack schemes, is defined by

$$d = \frac{n}{\log_2\{\max(a_1, a_2, \ldots, a_n)\}}.$$

Lagarias and Odlyzko introduced the attack for solving general low-density knapsacks [5)]. The latter attack was improved by Coster et al. [6)]. This attack would solve knapsack PKCs when the density $d < 0.9408$. This type of attacks is called the low-density attack.

Some trapdoors in knapsack schemes using modular multiplication(modular knapsack PKCs) has been proposed. Merkle and Hellman proposed the superincreasing sequence as the trapdoor [2)]. Kasahara and Murakami introduced a new trapdoor in modular knapsack PKC [7)].

*Division of Electronics and Communication Engineering, Graduate School of Engineering, Osaka Electro-Communication University

†Department of Telecommunications and Computer Networks, Faculty of Information and Communication Engineering, Osaka Electro-Communication University

2009 年 6 月 25 日受理

The trapdoor uses a even-odd check in shifted values of the sequence. Nasako and Murakami proposed the method of combining different trapdoors [8]. From the proposal of this method, different trapdoors in modular knapsack schemes can be arbitrarily combined together. The complexity of the secret sequence is made higher by combining some trapdoor sequences. So, it is valuable to consider new trapdoors which can be used in modular knapsack schemes.

In this paper, first, we shall propose a new trapdoor in modular knapsack PKC. This trapdoor uses some products of bases. Next, we shall propose a new modular knapsack PKC which use a proposed trapdoor. The trapdoor sequence proposed in this paper has a remarkable feature that plural bits can be decrypted at once. We shall also propose a special cases of the proposed scheme. Special cases are very simple schemes. Then, we shall discuss security against exhaustive search and low-density attack.

In the next section, we shall give some notations and some algorithms used in this paper. In Sect.2, we shall propose a new trapdoor and a modular knapsack scheme based on the proposed trapdoor. In Sect.4, we shall discuss the security of the proposed scheme. In the final section, we concludes this paper.

2 Proposed Trapdoor

In this section, we shall propose a trapdoor sequence and describe how to generate the proposed trapdoor sequence. In the proposed scheme, an n-dimensional plaintext message $\boldsymbol{m}$ is divided into l_i-dimensional divided plaintext messages $\boldsymbol{m}_i$ for $i = 1, 2, \ldots, t$.

2.1 Notation

$\mathbb{Z}_n$: $\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$;

$\boldsymbol{s} \in \mathbb{Z}^n$: secret key;

$\boldsymbol{a} \in \mathbb{Z}^n$: public key;

t : number of bases;

$b_i \in \mathbb{Z}$: the i-th base;

u : minimum bit size of random numbers;

$\boldsymbol{m} \in \{0,1\}^n$: plaintext message;

l_i : bit length of the i-th divided plaintext message;

$\boldsymbol{m}_i \in \{0,1\}^{l_i}$: the i-th divided plaintext message;

$M_i \in \mathbb{Z}_{2^{l_i}}$: integer message corresponding to $\boldsymbol{m}_i$;

$C \in \mathbb{Z}$: ciphertext;

$I \in \mathbb{Z}$: intermediate message;

2.2 Generation Algorithm G

1. Decide the system parameter t and l_i for $i = 1$ to t. Let $n = \sum_{k=1}^{t} l_k$, $L_i = \sum_{k=1}^{i} l_k$.
2. Generate the bases b_i, we assume that the following relation hold:

$$\lfloor \log_2 b_i \rfloor = l_i.$$

3. Decide the system parameter u and generate $\boldsymbol{s}$ from b_i as follows:
   $k = 1$,
   $B = 1$,
   For $i = 1$ to t do
     For $j = 1$ to l_i do
       Generate $(u + n - L_{i-1})$ bit integer c_{ij} such that $c_{ij} \equiv 2^{j-1} \pmod{b_i}$,
       $s_k = c_{ij}B$,
       $k \leftarrow k + 1$,
     done,
     $B \leftarrow B \times b_i$,
   done.

2.3 Proposed Scheme

In this subsection, we shall propose a new modular knapsack PKC which uses a proposed sequence as the trapdoor.

[Key Generation]

The keys of the proposed scheme are the followings:

Public key : $n, \boldsymbol{a}$.

Secret key : $\boldsymbol{s}, \{b_i\}, \{c_{11}, \cdots, c_{1l_1}, c_{21}, \cdots, c_{2l_2}, \cdots, c_{t1}, \cdots, c_{tl_t}\}$,
$(w_1, N_1), \ldots, (w_e, N_e), e$.

Bob creates a public key and a corresponding secret key by doing the following:

1. Generate the secret key $\boldsymbol{s}$ with Algorithm $\mathtt{G}$.
2. Compute the public key $\boldsymbol{a} \in \mathbb{Z}^n$ as follows:
   (1) $\boldsymbol{a} = \boldsymbol{s}$.
   (2) For $i = 1$ to e do
     i. Choose the modulus N_i such that $N_i > \sum_{k=1}^{n} a_k$.
     ii. Select a random integer w_i such that $\gcd(w_i, N_i) = 1$.
     iii. Compute the next $\boldsymbol{a}$ by the modular multiplication (w_i, N_i):

$$\boldsymbol{a} \leftarrow w_i \boldsymbol{a} \pmod{N_i}.$$

   done.

It should be noted that a_k is overwritten.

[Encryption]

Alice encrypts a message $\boldsymbol{m} = (m_1, m_2, \ldots, m_n) \in \{0, 1\}^n$ into the ciphertext $C \in \mathbb{Z}$ by the following:

1. Calculate the ciphertext C:

$$C = \boldsymbol{a}\boldsymbol{m}^T,$$

   where $\boldsymbol{m}^T$ denotes the transposed of $\boldsymbol{m}$.

[Decryption]

Let us define by the intermediate message I be $I = \boldsymbol{s}\boldsymbol{m}^T$.

Bob decrypts the plaintext message $\boldsymbol{m} \in \{0,1\}^n$ from the ciphertext $C \in \mathbb{Z}$ by doing the following:

1. Compute the intermediate message $I \in \mathbb{Z}$ over $\mathbb{Z}$:

   (1) $I = C$.

   (2) For $i = e$ downto 1 do

   $$I \leftarrow w_i^{-1} I \pmod{N_i},$$

   done.

2. Obtain the integer messages $M_i \in \mathbb{Z}_{2^{l_i}}$ as follows:
   $l = 0$.
   For $i = 1$ to t do

   $$M_i = I \bmod b_i,$$

   $$I \leftarrow (I - M_i)/b_i,$$

   done.

3. Obtain the plaintext message $\boldsymbol{m} = [\boldsymbol{m}_1 \mid \boldsymbol{m}_2 \mid \cdots \mid \boldsymbol{m}_t]$ by regarding M_i as a binary vector $\boldsymbol{m}_i \in \{0,1\}^{l_i}$.

2.4 Density

The density d of the proposed scheme satisfies the following inequality:

$$\begin{aligned} d &> \frac{n}{\lceil \log_2 N_e \rceil}, \\ &> \frac{n}{n + u + e \log_2 n}, \end{aligned}$$

where we let $b_i = 2^{l_i} + \varepsilon_i$ and $1 \ll \varepsilon_i \ll 2^{l_i}$.

3 Special Cases

In this section, we shall propose two special cases of the proposed schemes for practical realization.

3.1 Special Case I

In this subsection, we shall describe the scheme of the special case that $t = 1$.

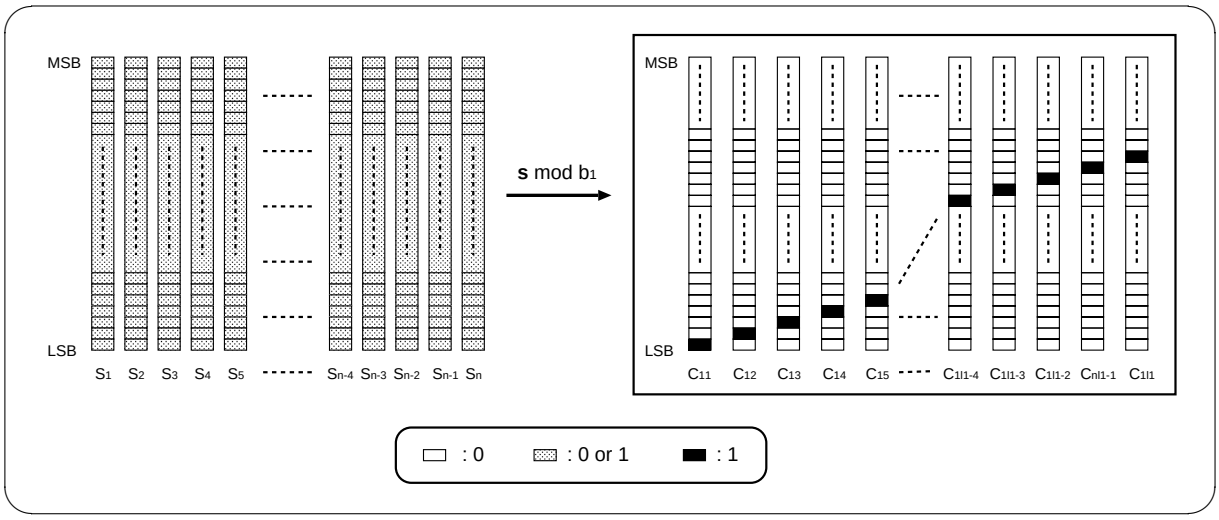

Figure 1: Trapdoor of the special case I

[Key Generation]

The keys of the special case I are the followings:

Public key : $n, \boldsymbol{a}$.

Secret key : $\boldsymbol{s}, b_1, \{c_{11}, \cdots, c_{1l_1}\}, (w_1, N_1), \ldots, (w_e, N_e), e$.

Bob creates a public key and a corresponding secret key by doing the following:

1. Generate the secret key $\boldsymbol{s}$ for $t = 1$ and $l_1 = n$.
2. Compute the public key $\boldsymbol{a} \in \mathbb{Z}^n$ as follows:
   (1) $\boldsymbol{a} = \boldsymbol{s}$.
   (2) For $i = 1$ to e do
      i. Choose the modulus N_i such that $N_i > \sum_{k=1}^{n} a_k$.
      ii. Select a random integer w_i such that $\gcd(w_i, N_i) = 1$.
      iii. Compute the next $\boldsymbol{a}$ by the modular multiplication (w_i, N_i):

$$\boldsymbol{a} \leftarrow w_i \boldsymbol{a} \pmod{N_i},$$

   done.

Figure 1 shows the trapdoor of the special case I.

[Encryption]

Alice encrypts a message $\boldsymbol{m} \in \{0, 1\}^n$ into the ciphertext $C \in \mathbb{Z}$ by the following:

1. Calculate the ciphertext C:

$$C = \boldsymbol{a}\boldsymbol{m}^T.$$

[Decryption]

Let I is intermediate message such that $I = \boldsymbol{s}\boldsymbol{m}^T$.

Bob decrypts the message $\boldsymbol{m} \in \{0,1\}^n$ from the ciphertext $C \in \mathbb{Z}$ by doing the following:

1. Compute the intermediate message $I \in \mathbb{Z}$ over $\mathbb{Z}$:

   (1) $I = C$.

   (2) For $i = e$ downto 1 do

   $$I \leftarrow w_i^{-1} I \pmod{N_i},$$

   done.

2. Obtain the integer messages $M_1 \in \mathbb{Z}_{l_1}$ as follows:

   $$M_1 = I \bmod b_1.$$

3. Obtain the plaintext message $\boldsymbol{m} = \boldsymbol{m}_1$ by regarding M_1 as a binary vector $\boldsymbol{m}_1 \in \{0,1\}^{l_1}$.

3.2 Special case II

In this subsection, we shall describe the scheme of the special case that $t = 2$.

[Key Generation]

The keys of the special case II are the followings:

Public key : $n, \boldsymbol{a}$.

Secret key : $\boldsymbol{s}, \{b_1, b_2\}, \{c_{11}, \cdots, c_{1l_1}, c_{21}, \cdots, c_{2l_2}\}, (w_1, N_1), \ldots, (w_e, N_e), e$.

Bob creates a public key and a corresponding secret key by doing the following:

1. Generate the secret key $\boldsymbol{s}$ for $t = 2$.
2. Compute the public key $\boldsymbol{a} \in \mathbb{Z}^n$ as follows:

   (1) $\boldsymbol{a} = \boldsymbol{s}$.

   (2) For $i = 1$ to e do

   i. Choose the modulus N_i such that $N_i > \sum_{k=1}^{n} a_k$.

   ii. Select a random integer w_i such that $\gcd(w_i, N_i) = 1$.

   iii. Compute the next $\boldsymbol{a}$ by the modular multiplication (w_i, N_i):

   $$\boldsymbol{a} \leftarrow w_i \boldsymbol{a} \pmod{N_i},$$

   done.

Figure 2 shows trapdoor of the special case II.

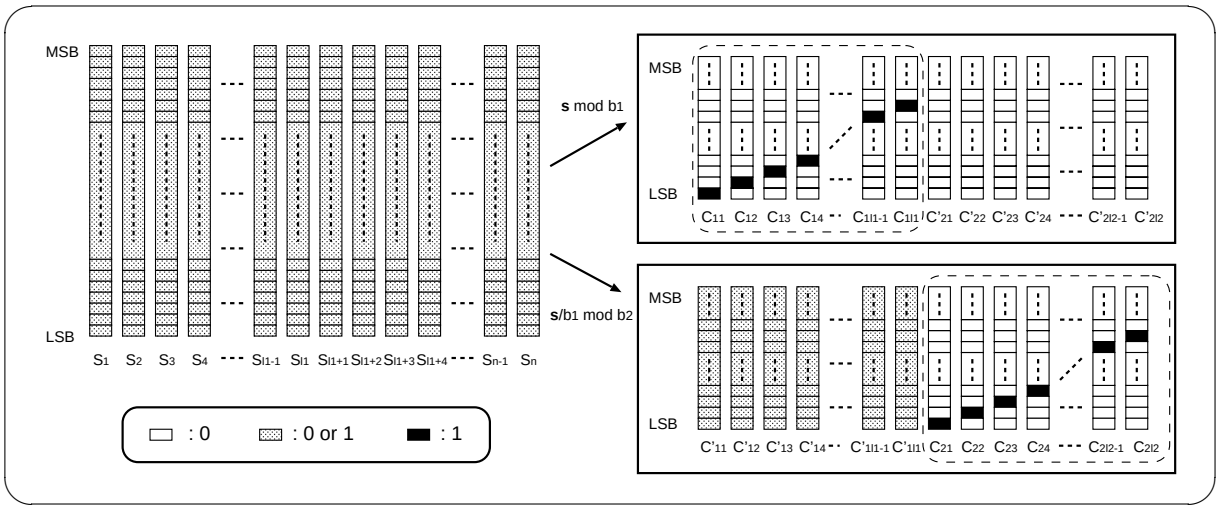

Figure 2: Trapdoor of the special case II

[Encryption]

Alice encrypts a message $\boldsymbol{m} \in \{0,1\}^n$ into the ciphertext $C \in \mathbb{Z}$ by the following:

1. Calculate the ciphertext C:

$$C = \boldsymbol{a}\boldsymbol{m}^T,$$

where $\boldsymbol{m}^T$ denotes the transposed of $\boldsymbol{m}$.

[Decryption]

Let I is intermediate message such that $I = \boldsymbol{s}\boldsymbol{m}^T$.

Bob decrypts the message $M \in \mathbb{Z}_{2^n}$ from the ciphertext $C \in \mathbb{Z}$ by doing the following:

1. Compute the intermediate message $I \in \mathbb{Z}$ over $\mathbb{Z}$:

   (1) $I = C$.

   (2) For $i = e$ downto 1 do

$$I \leftarrow w_i^{-1} I \pmod{N_i},$$

   done.

2. Obtain the integer message $M_i \in \mathbb{Z}_{2^{l_i}}$ as follows:
   For $i = 1$ to t do

$$M_i = I \bmod b_i,$$

$$I \leftarrow (I - M_i)/b_i,$$

   done.

3. Obtain the plaintext message $\boldsymbol{m} = [\boldsymbol{m}_1 \mid \boldsymbol{m}_2]$ by regarding M_1 and M_2 as binary vectors $\boldsymbol{m}_1 \in \{0,1\}^{l_1}$ and $\boldsymbol{m}_2 \in \{0,1\}^{l_2}$, respectively.

4 Security Discussions

In this section, we shall discuss about the security of the proposed scheme.

4.1 Security of Secret Key

Several attacks of computing the secret key from the public key are proposed on the knapsack PKC. Shamir's attack [3] and Adleman's attack [4] are typical attacks of this type. Shamir's attack and Adleman's attack can break the knapsack PKC when the public key is made from a superincreasing sequence with the modular multiplication. However, the proposed trapdoor dose not use the superincreasing sequence. Thus, the proposed scheme can be secure against Shamir's attack and Adleman's attack. However, there may exist an effective attack for the secret key of the proposed scheme. We would like to continue discussions on the security of the secret key against this type of attacks at a later day.

4.2 Security against Exhaustive Search

The exhaustive search is an attack by searching message at all possibilities. It requires a great investment of time to search for 64 bits even by the latest computers. We strongly recommend that $n \geq 64$ and $u \geq 64$ in order to be secure against the exhaustive search.

4.3 Security against Low-Density Attack

Low-density attack works effectively low-density knapsack PKCs irrespective of the trapdoors. This attack reduce the subset sum problem to that of finding a short vector in a lattice(SVP) when the density of subset sum problem is less than 0.9408 with SVP oracle [6]. LLL algorithm is known to be a practical algorithm to compute the shortest vector in lattice. However, this algorithm does not work well when $n > 500$. Thus, LLL algorithm is no practical algorithm to solve the subset sum problem of large dimension. In the proposed scheme we recommend $n > 1000$.

4.3.1 Computer experiment of Low-Density Attack

We show the result of computer experiment of the low-density attack to the special cases of the proposed scheme. Coster et al. proposed the low-density attack which can solve the subset sum problem of the density less than 0.9408 [6] with a lattice reduction by using the matrix:

$$A = \begin{pmatrix} 1 & 0 & \dots & 0 & -\lambda a_1 \\ 0 & 1 & \dots & 0 & -\lambda a_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & -\lambda a_n \\ -1/2 & -1/2 & \dots & -1/2 & \lambda C \end{pmatrix},$$

where $\lambda > \sqrt{2n}$. The problem of calculating the plaintext message can be reduced into that of solving finding short vectors in the lattice spanned by row vectors of $L(A)$.

This computer experiment is done as follows:

1. Set the number of partitions $t = 1, 2$.
2. Set the minimum bit size of random numbers $u = 64$.
3. Generate 100 public keys for $n = 24, 26, 28, \dots, 200$.

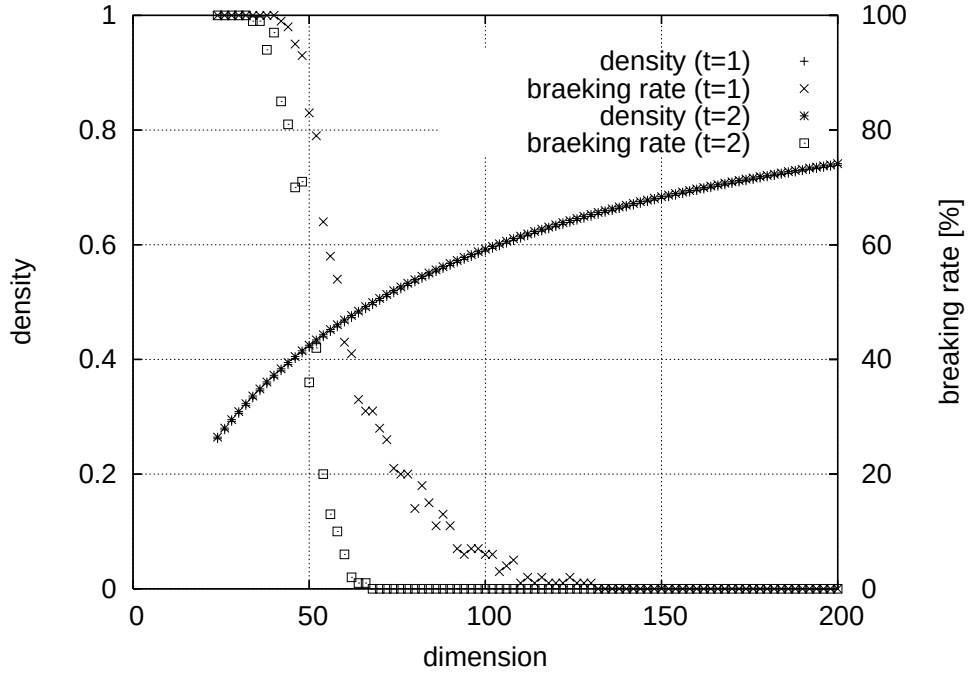

Figure 3: Density and Breaking rate

4. Generate 100 plaintexts for each public key.

5. Low-density attack against the each ciphertext with LLL algorithm [9].

Figure 3 shows the density and breaking rate of the proposed scheme when $e = 1$, $t = 1, 2$. In Fig.3, the horizontal and the vertical axises show the dimension n of the plaintext message and the density and breaking rate, respectively. It is seen that any plaintext message can not be found when $n > 130$ from Fig.(3). From the result, the proposed scheme would be able to be secure against the low-density attack when n is sufficiently large.

5 Conclusion

In this paper, we have proposed a new trapdoor which uses the some products of bases. We also have proposed a modular knapsack PKC which based on the proposed trapdoor. The trapdoor sequence has a remarkable feature that plural bits can be decrypted at once. We have also proposed two special cases of the proposed scheme. Moreover, we have discussed the security against the exhaustive search and the low-density attack. As the result, we can conclude that the proposed scheme is sufficiently secure.

References

1) P.W.Shor: "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," SIAM J. Comput. 26, 5, pp.1484–1509(1997).

2) R.C.Merkle and M.E.Hellman: "Hiding information and signatures in trapdoor knapsacks," IEEE Transactions on Information Theory 24, 5, pp.525–530(1978).

3) A.Shamir: “A polynomial time algorithm for breaking the basic Merkle-Hellman cryptosystem,” IEEE Symposium on Foundations of Computer Science, pp.145–152(1982).

4) L.M.Adleman: “On breaking the titrated Merkle-Hellman public-key cryptosystem,” Plenum Press. Crypto’82, pp.303–308(1982).

5) J.C.Lagarias and A.M.Odlyzko: “Solving low-density subset sum problems,” J. ACM 32, 1, pp229–246(1985).

6) M.J.Coster, B.A.LaMacchia, A.M.Odlyzko and C.P.Schnorr: “An improved low-density subset sum algorithm,” Lecture Notes in Computer Science, 547, pp.54–67(1991).

7) M.Kasahara and Y.Murakami: “New public key cryptosystems and the application,” Technical Report of IEICE, ISEC99–55, pp.21–28(1999).

8) Y.Murakami and T.Nasako: “A new class of knapsack public-key cryptosystems using modular multiplication,” The 1st joint workshop on information security, pp.351–354(2006).

9) A.Lenstra, H.L.Jr and L.Lovasz: “Factoring polynomials with rational coefficients,” Mathematische Annalen 261, 4, pp.515–534(1982).